

The Command-line

Part 1: Concepts and Demonstration

Today's plan

Speaker: Daniel Flaum, an undergraduate computer science student at SIUE.

- We'll introduce the command-line
 - What it is
 - How it fits into a computer system
- And demonstrate it
 - Basic command grammar
 - Commands to manipulate files
 - Examples of command-line features
 - A case study of command-line scripting
- There's a handout! It includes:
 - Extra details
 - Book recommendations

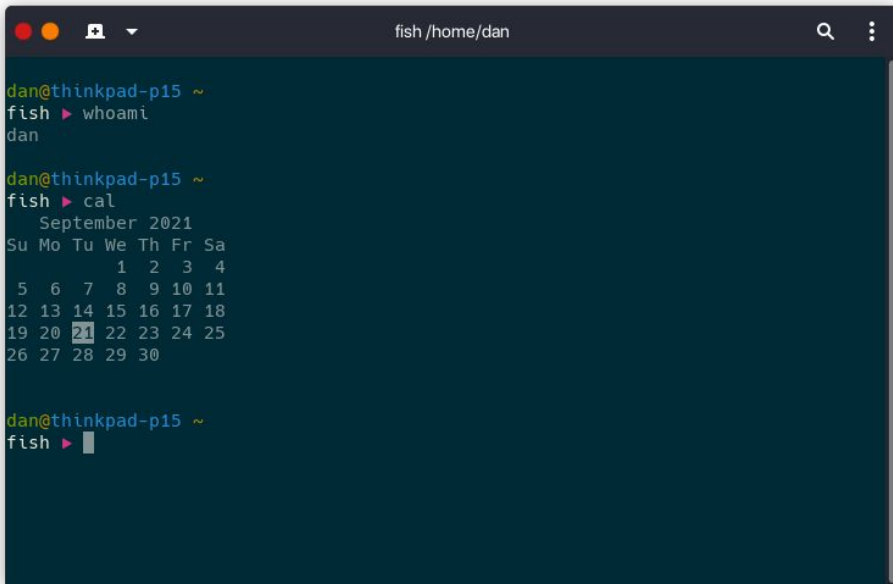
This is the first of two parts.

- Part 2 will give you hands-on experience
 - Access to a command-line on our server through your web browser
 - Explore a digital humanities application
- Same time, same place
 - 10:30a, November 12th, 2021

A recording of today's talk and the handout will be made available at **iris.siue.edu/workshops**.

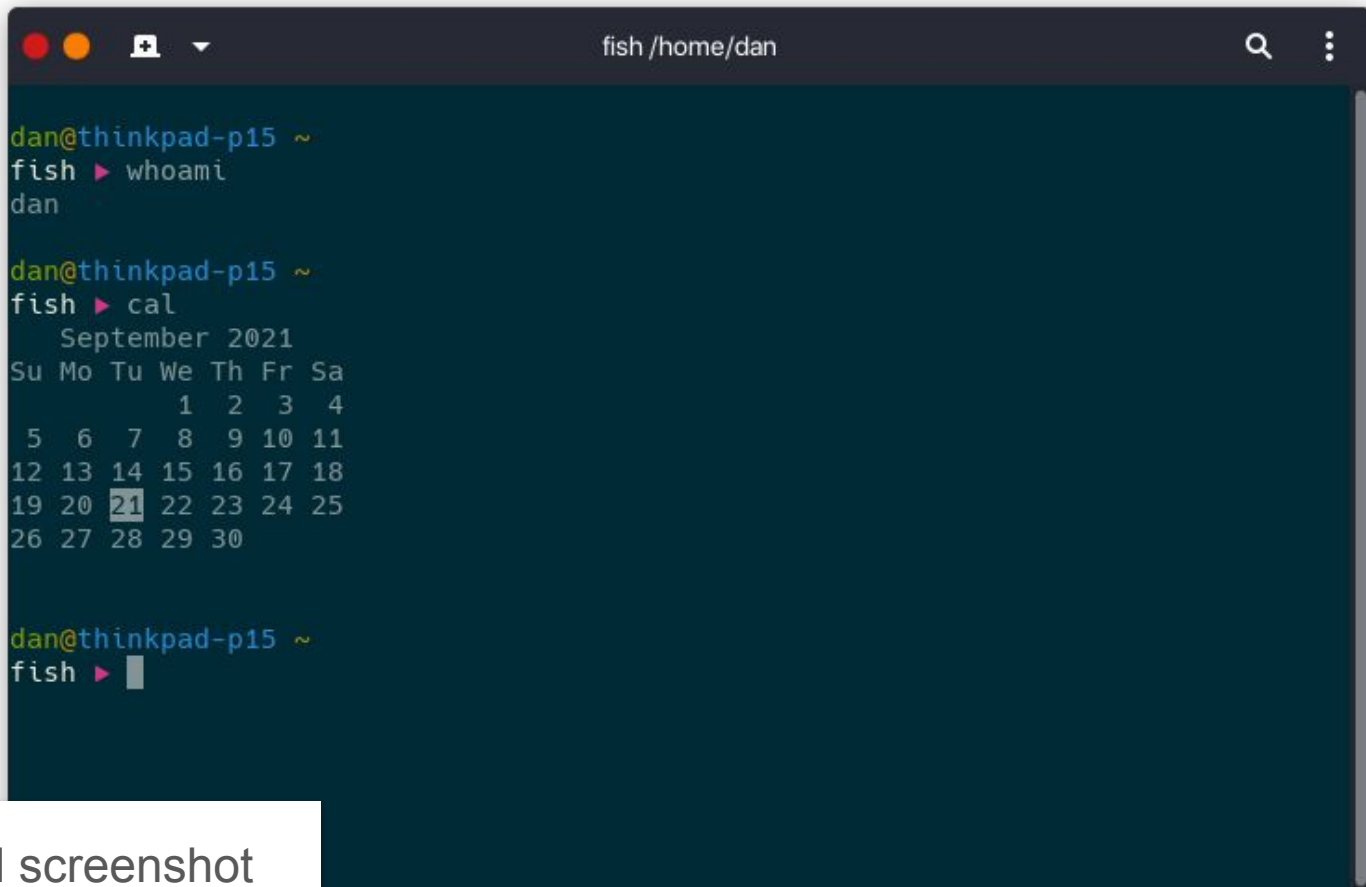
What is the command-line?

- Correction: what is a command-line?
(There's more than one.)
- A primitive way of using a computer
 - The user types out and submits *commands*, one after another.
 - *Commands* launch *programs* to do useful work.
 - Programs usually display *output* and then terminate.
- A predecessor of the modern graphical desktop
 - Dates back to the 1960s.
 - Uses only text, which is much simpler for programmers to work with.
 - Continues to be an indispensable part of all kinds of computers.



A screenshot of a terminal window with a dark blue background. The window title bar shows 'fish /home/dan'. The terminal displays the following text:

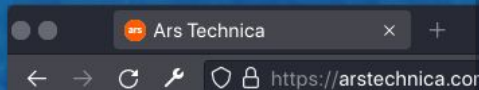
```
dan@thinkpad-p15 ~  
fish ▶ whoami  
dan  
  
dan@thinkpad-p15 ~  
fish ▶ cal  
      September 2021  
Su Mo Tu We Th Fr Sa  
      1  2  3  4  
 5  6  7  8  9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29 30  
  
dan@thinkpad-p15 ~  
fish ▶
```



A terminal window titled "fish /home/dan" with standard macOS window controls. The terminal shows the following sequence of commands and output:

```
dan@thinkpad-p15 ~  
fish ▶ whoami  
dan  
  
dan@thinkpad-p15 ~  
fish ▶ cal  
    September 2021  
Su Mo Tu We Th Fr Sa  
      1  2  3  4  
  5  6  7  8  9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29 30  
  
dan@thinkpad-p15 ~  
fish ▶ █
```

Enlarged screenshot



Facebook's late

"I" [red dot] [orange dot] [blue dot] [white dot]

```
dan@thinkpad-p15 ~  
fish ▶ whoami  
dan
```



```
dan@thinkpad-p15 ~  
fish ▶ cal
```

```
September 2021  
Su Mo Tu We Th Fr Sa  
Sc 1 2 3 4  
BE 5 6 7 8 9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29 30
```

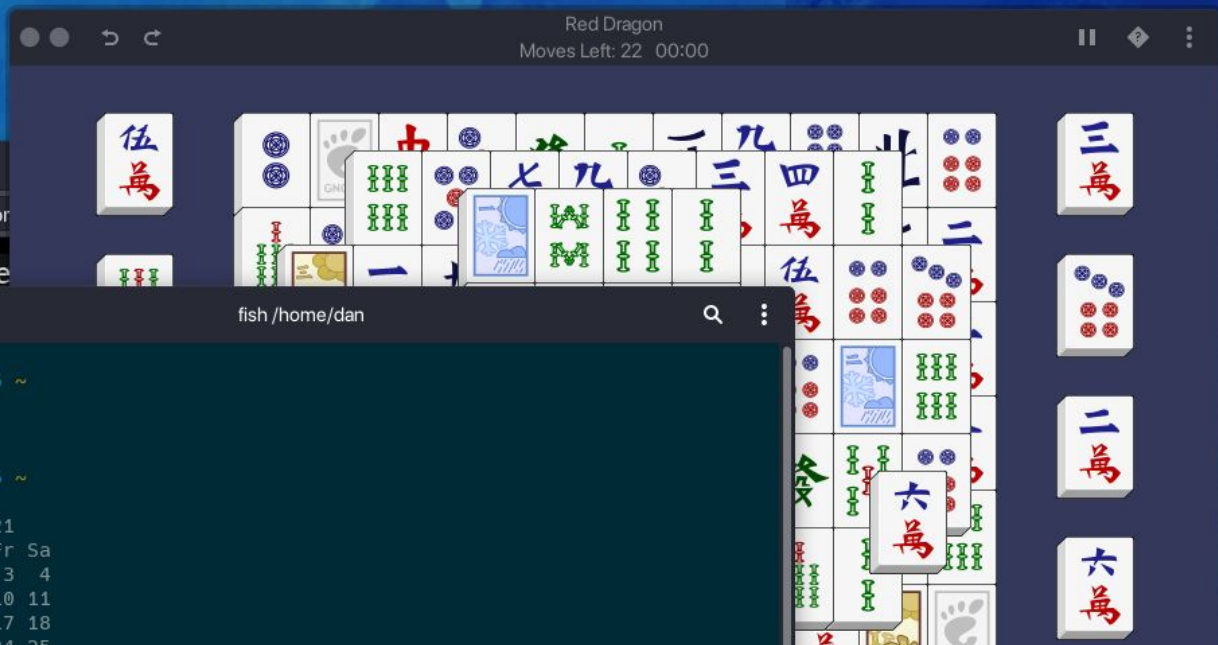
LATEST STORIES CONT



```
dan@thinkpad-p15 ~  
fish ▶ █
```

```
e:  
Go  
RC
```

Updated PC Health Check app will
run Windows 11



A screenshot of the a command-line running in a familiar graphical desktop environment on the presenter's computer.

It appears as colored text inside a normal graphical window alongside a Web browser and a game of Mahjong solitaire.

```

Fedora 34 (Workstation Edition)
Kernel 5.13.13-200.fc34.x86_64 on an x86_64 (tty2)

thinkpad-p15 login: dan
Password:
Last login: Tue Sep 21 19:23:50 on tty2
-bash: /home/dan/.cargo/env: No such file or directory
-bash: /home/dan/.cargo/env: No such file or directory
-bash: /home/dan/.cargo/env: No such file or directory

dan@thinkpad-p15 ~
165 $ whoami
dan

dan@thinkpad-p15 ~
166 $ cal
      September 2021
Su Mo Tu We Th Fr Sa
                1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30

dan@thinkpad-p15 ~
167 $

```

A photograph of the same command-line, but outside the graphical environment.

It appears as colored text on a solid background occupying the entire screen. The text shows the user first logging in, then running a few commands.

Fedora 34 (Workstation Edition)

Kernel 5.13.13-200.fc34.x86_64 on an x86_64 (tty2)

thinkpad-p15 login: dan

Password:

Last login: Tue Sep 21 19:23:50 on tty2

-bash: /home/dan/.cargo/env: No such file or directory

-bash: /home/dan/.cargo/env: No such file or directory

-bash: /home/dan/.cargo/env: No such file or directory

These error messages are my fault, ignore them

```

dan@thinkpad-p15 ~
165 $ whoami
dan

```

```

dan@thinkpad-p15 ~
166 $ cal
      September 2021
Su Mo Tu We Th Fr Sa
                1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30

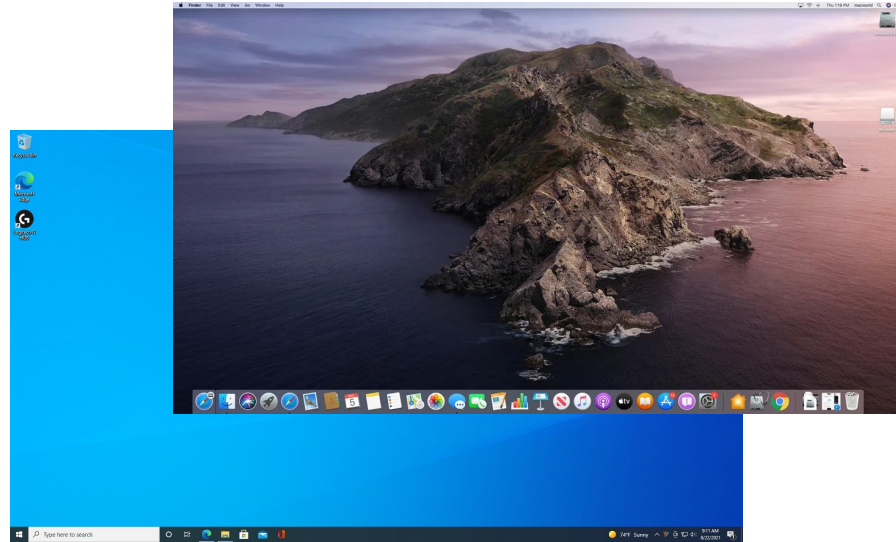
```

```

dan@thinkpad-p15 ~
167 $

```

Where does the command-line fit in?

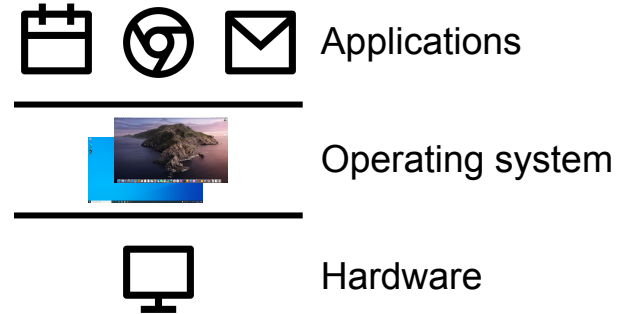


Consider the computer as a whole

Three layers:

- The physical hardware
- The operating system
(which must be compatible with the hardware)
- The applications
(which must be compatible with the OS)

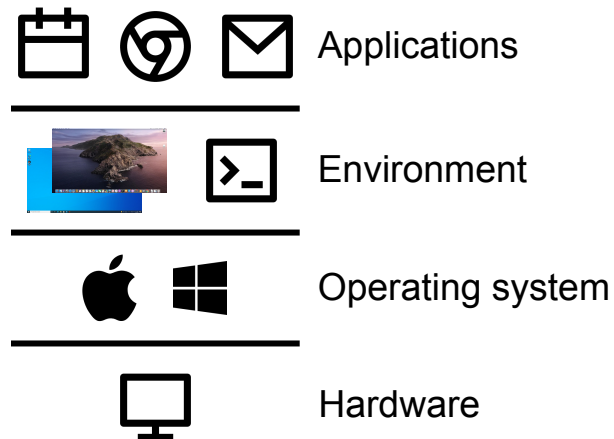
Humans use the OS to start and navigate among the applications, where actually useful work gets done.



Adding the command-line requires adding a fourth layer

- The physical hardware
- The operating system
(which must be compatible with the hardware)
- **The environment**
(which must be compatible with the OS)
- The applications
(which must be compatible with the OS)

Humans use the environment, *not* the OS, to start and navigate among the applications, where actually useful work gets done.



The environment is separate from the OS

- It serves as the user interface for the OS. It's a habitat for humans.
- When a computer doesn't need to offer an user interface the OS can run *headless*, with no environment at all.
- When you imagine Windows or macOS, you're actually visualizing the environments that Microsoft and Apple provide with them.
- There is no *technical* reason why there cannot be more than one environment.

B-slides

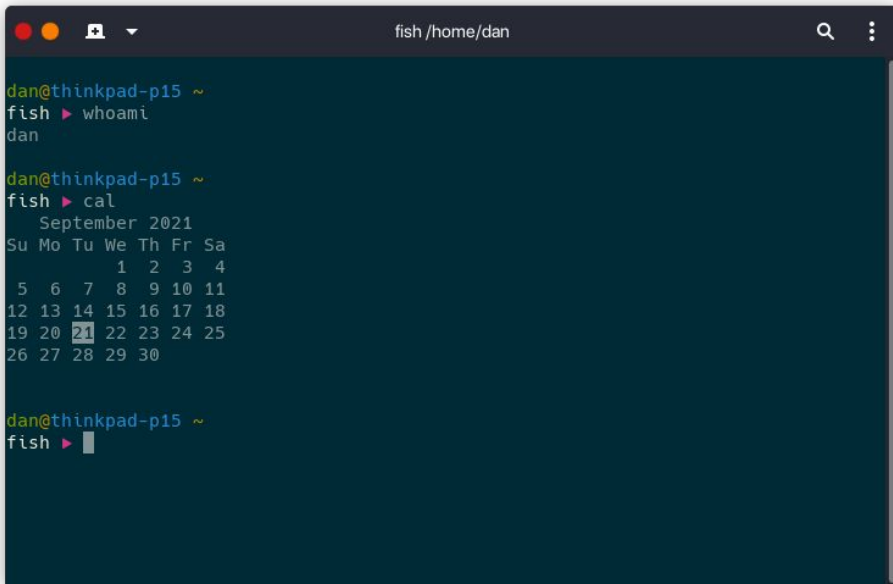
Following are early draft slides. They weren't used during the talk, but you might still find them useful.

What is the command-line?

In a nutshell, a way of using a computer by entering textual commands one after another which cause programs to run and do useful work.

It dates back to the 1960s and is still widely used today because it's much, *much* easier to write and maintain programs that use text than ones that use graphics.

But it's also often intimidating to new users because they can't see what's happening.



```
fish /home/dan

dan@thinkpad-p15 ~
fish ► whoami
dan

dan@thinkpad-p15 ~
fish ► cal
  September 2021
Su Mo Tu We Th Fr Sa
    1  2  3  4
  5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30

dan@thinkpad-p15 ~
fish ►
```

We're going to start by putting the
command-line in context.

The operating system

Every computer comes with an operating system. The OS is a complex collection of cooperating software programs, some of them with special functions and properties.

The job of the OS is to make life easier both on the user (you) and on software programmers by simplifying the nitty gritty details of your computer's hardware.

Two operating systems dominate the market for most people: Microsoft Windows and Apple macOS.

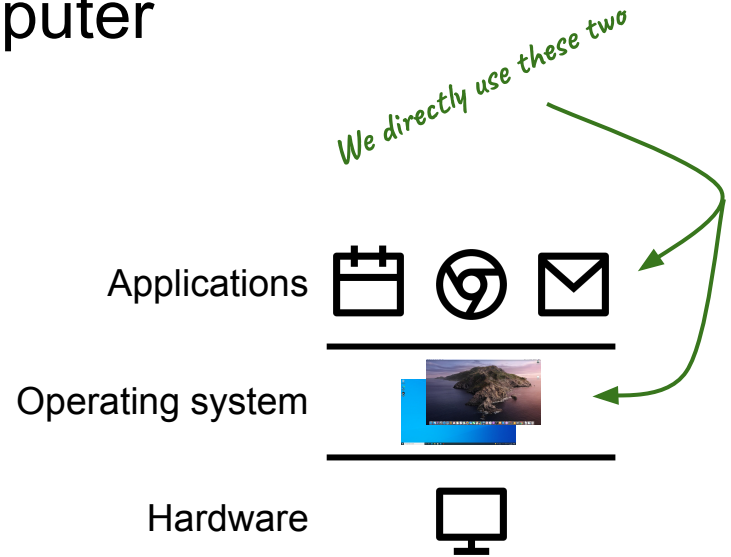


The typical understanding of a computer

Most people understand the computer as three layers, each depending on the one below.

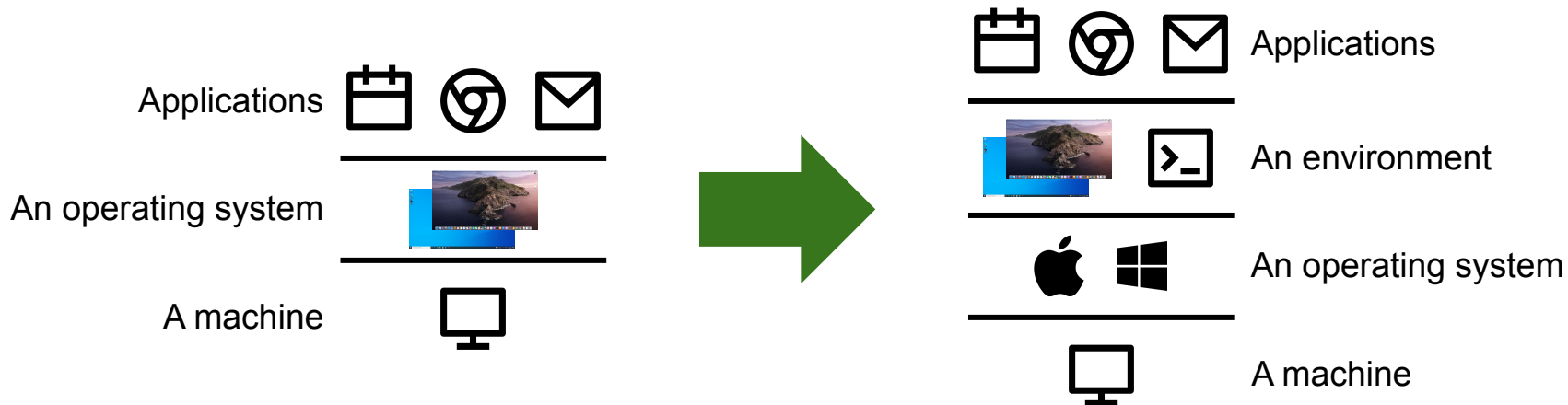
Applications like Microsoft Word or Apple iMovie are on top, and they need the right OS to run. Operating systems are in the middle, and they need the right hardware to run.

So which OS you can run depends on what machine you have, and what apps you can use depends on what OS you run.



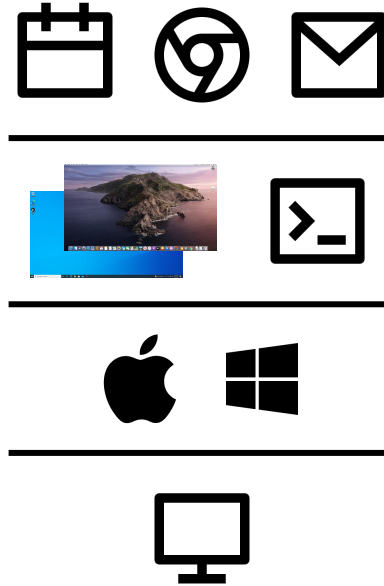
Where the command-line fits in

The command-line requires a subtler understanding of the computer



Where the command-line fits in

The command-line requires a subtler understanding of the computer



The command-line goes here

The environment

The **environment** is a software system that provides a consistent, overarching user interface inside which applications can be opened, used, hidden, and terminated.

When you imagine Windows or macOS, the picture that comes to mind is of the environment, not the actual OS.

The graphical environment

In our usual day-to-day computer use, we generally work inside the **graphical environment** provided by Windows or macOS.

The environment provides a place for programs to draw **graphical user interfaces** made of buttons, icons, menus, and so on. Each program gets at least one **window**, which is meant to be a virtual computer screen.

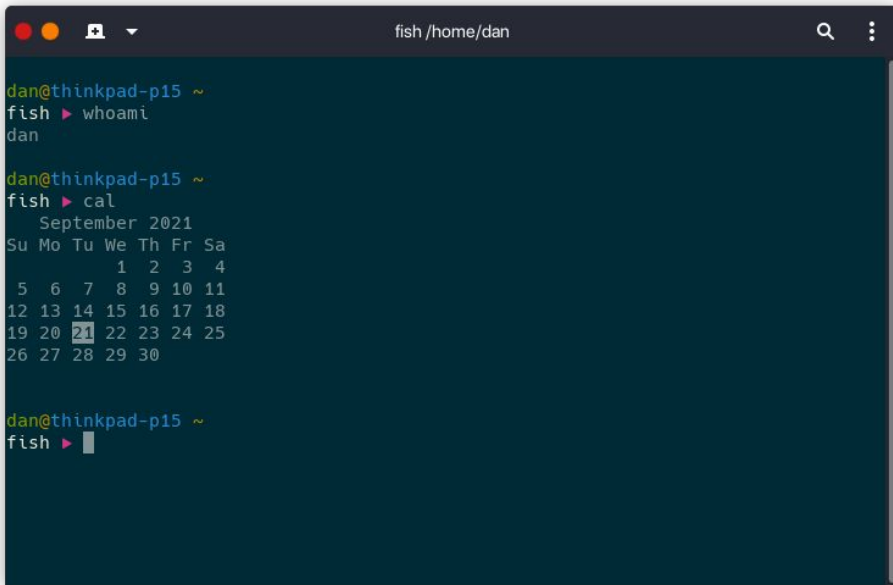
The graphical environment also provides overarching features, like the taskbar and start menu in Windows, or the menu bar and dock in macOS.



The textual environment

The command-line is a **textual environment**. It uses plain text, all of the same font and all of the same size. The text can be different colors, and maybe it can bold, italic, or underlined, but that's it.

Unlike the desktop environment, it doesn't pursue any metaphor at all. But it still does the job of an environment: "a consistent, overarching user interface inside which applications can be opened, hidden, and terminated."

A screenshot of a terminal window with a dark blue background. The window title bar at the top shows three colored circles (red, yellow, green) on the left, the text 'fish /home/dan' in the center, and search and menu icons on the right. The terminal content shows a user named 'dan' at a machine named 'thinkpad-p15'. The user enters the command 'whoami', and the output is 'dan'. Then, the user enters 'cal', and the output is a calendar for September 2021. The calendar is displayed in a text-based grid format. Finally, the user enters a command that results in a single block character being printed.

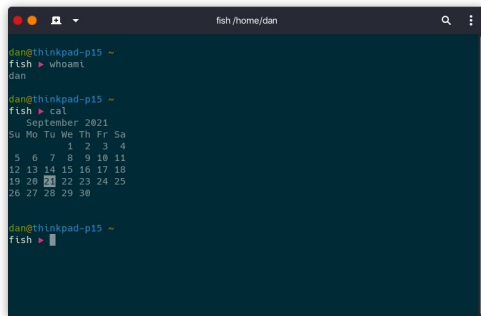
```
dan@thinkpad-p15 ~  
fish ▶ whoami  
dan  
  
dan@thinkpad-p15 ~  
fish ▶ cal  
      September 2021  
Su Mo Tu We Th Fr Sa  
          1  2  3  4  
 5  6  7  8  9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29 30  
  
dan@thinkpad-p15 ~  
fish ▶ █
```

The command-line

If we like our terminology to be nice and regular, then the command-line might be better called a textual environment, to complement the graphical environment. In practice, the command-line is often called the **shell** instead.

Like the graphical environment, the shell provides a place for programs to display their output. This place takes the form of a virtual, infinite sheet of paper (by design, since terminal displays were preceded by paper printouts).

The command-line also provides overarching features that correspond to things like the taskbar in Windows or the menu bar in macOS.



```
dan@thinkpad-p15 ~  
fish > whoami  
dan  
  
dan@thinkpad-p15 ~  
fish > cal  
      September 2021  
Su Mo Tu We Th Fr Sa  
  1  2  3  4  
  5  6  7  8  9 10 11  
 12 13 14 15 16 17 18  
 19 20 21 22 23 24 25  
 26 27 28 29 30  
  
dan@thinkpad-p15 ~  
fish > 
```

=

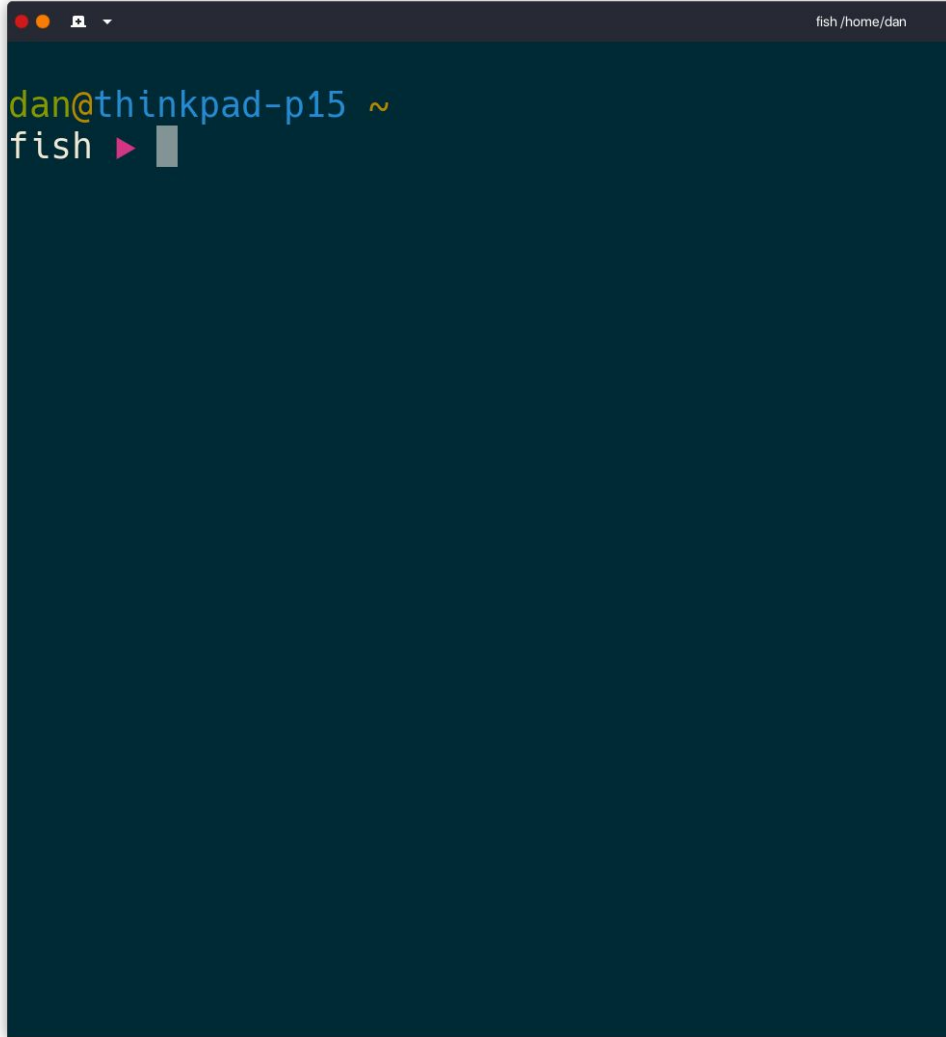


The command-line

Wikipedia has a long list of shells both obsolete and current. But there are only four you should know about.

On Windows, there are two: cmd and PowerShell. According to Microsoft, cmd is the old command-line, and you should put your effort into learning the newer PowerShell.

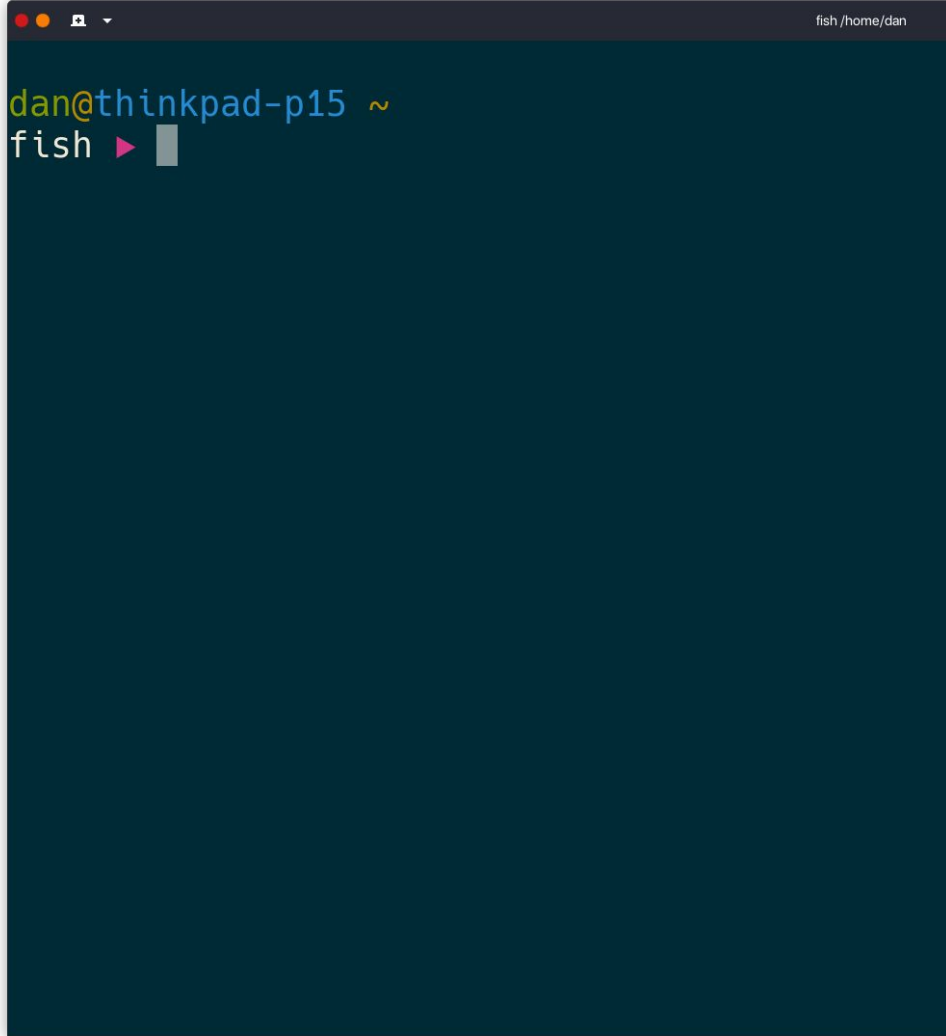
On macOS there is Zsh. Zsh is an independent software project--Apple doesn't own it.



The command-line\$

The fourth shell is fish, which is short for Friendly Interactive Shell. This is the presenter's choice because it offers many modern features that make the command-line easier to learn and use.

We'll practice with fish later, but know that if you're on Windows; you'll want to learn PowerShell and if you're on macOS you'll want to learn bash.



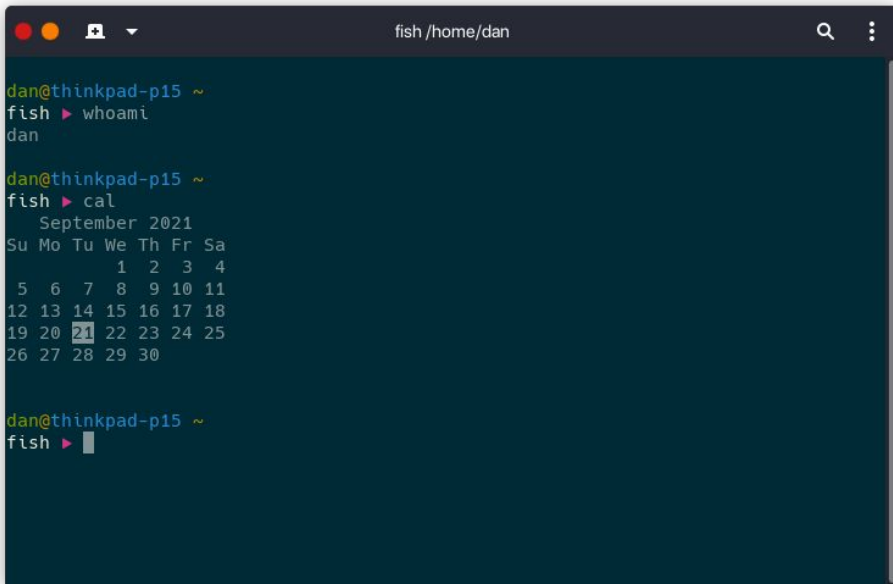
The terminal emulator

We don't want to leave our cozy graphical environment, so we most often access the command line by running a program called a **terminal emulator**.

Today we're used to personal computers which either sit on our lap or under our desk. But a few decades ago computers were much larger and much more expensive. They occupied dedicated rooms and were shared by many people.

To avoid users having to physically be in the room with the computer, users were given a **terminal**: a compact combination of a printer or display and keyboard that could be set up in any room and was connected to the computer by wires run throughout the building (or even the campus).

A terminal emulator is just like one of these terminals, except that it's virtual—it appears inside a graphical window, just like any other application you're used to.

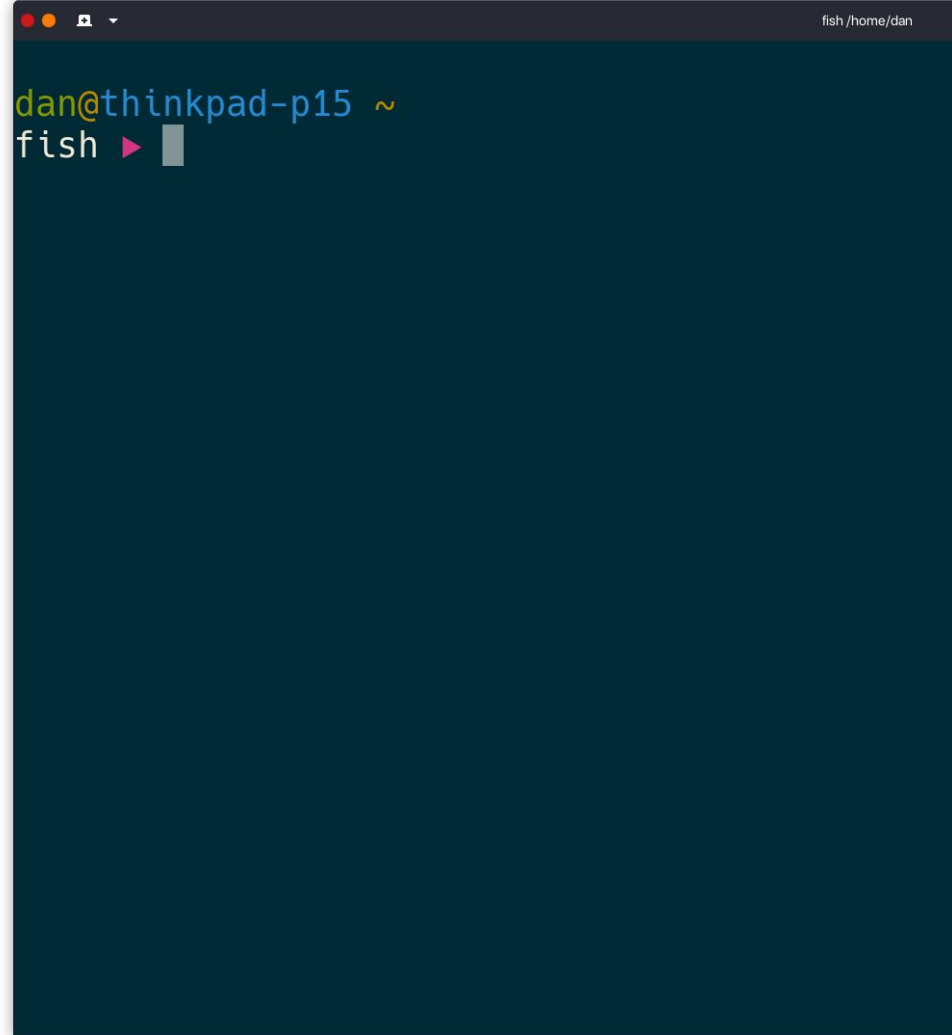


The image shows a terminal emulator window with a dark blue background. The window title bar at the top is dark grey and contains the text 'fish /home/dan' on the right side, along with standard window control icons (red, yellow, green buttons and a menu icon). The terminal content shows a user named 'dan' at a machine named 'thinkpad-p15' in the '~' directory. The user has entered the command 'whoami', which returned 'dan'. Then, the user entered 'cal', which displayed a calendar for September 2021. The calendar is a grid with days of the week as columns and dates as rows. The 21st is highlighted. Finally, the user entered a command that resulted in a blank line.

```
dan@thinkpad-p15 ~  
fish ► whoami  
dan  
  
dan@thinkpad-p15 ~  
fish ► cal  
      September 2021  
Su Mo Tu We Th Fr Sa  
    1  2  3  4  
  5  6  7  8  9 10 11  
12 13 14 15 16 17 18  
19 20 21 22 23 24 25  
26 27 28 29 30  
  
dan@thinkpad-p15 ~  
fish ►
```

The prompt

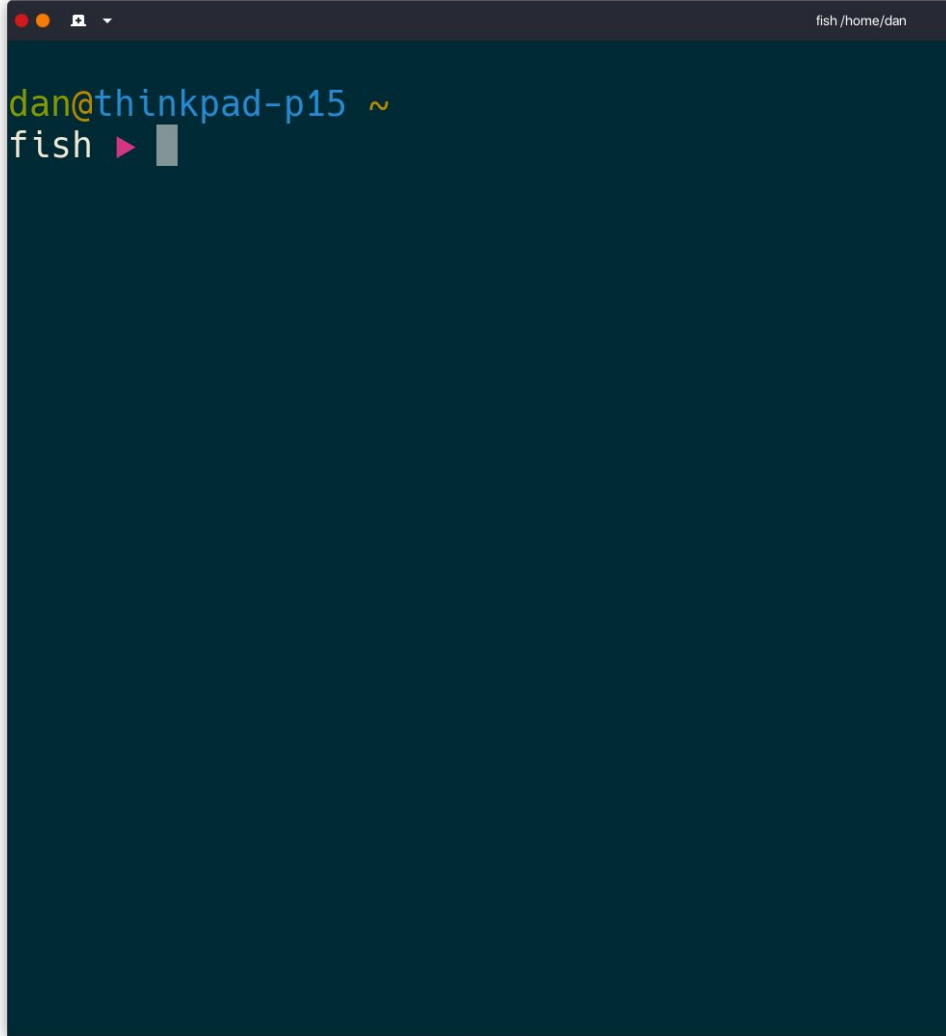
When the command-line is first opened the user is presented with the **prompt**, a block of text that provides a place to enter a **command**.

A screenshot of a terminal window with a dark blue background. The window title bar at the top shows standard macOS window controls (red, yellow, green buttons) and a dropdown menu. The text 'fish /home/dan' is visible in the top right corner of the window. The main content of the terminal shows the prompt 'dan@thinkpad-p15 ~' in a light blue font, followed by 'fish' in a light pink font, a right-pointing triangle, and a grey rectangular cursor block.

```
dan@thinkpad-p15 ~  
fish ► █
```

The prompt

Different command-lines come with different default prompts. Besides prompting, they usually include some vital information useful to have printed before every command. The presenter's includes the username of logged in user, the name of the computer, the current location in the filesystem (~), and the name of the command-line.



The prompt

Prompts are freely customizable, and many power users find it worthwhile to spend a bit of time learning how to arrange their own.

